Mudança de runlevels e desligamento ou reinicialização do sistema

Sumário

Índice de tabelas

Índice de Figuras

Capítulo 1

Mudança de Runlevels

1.1. Objetivos

- Conhecer os runlevels do sistema;
- Verificar o runlevel;
- Trocar o runlevel;
- Desligar e reiniciar o servidor em modo seguro.

1.2. Mãos a obra

Você já deve ter se perguntado como o Sistemas Linux fazem para iniciar seus programas. A resposta é simples, basta apenas colocar o script dentro do diretório do runlevel correspondente. Vamos ver como isso é possível e como podem ser feitas as mudanças de runlevel.

Mas o que é um runlevel?

Um runlevel é o nível de inicialização do sistema. Em algumas distribuições baseadas em Debian e RedHat, utilizam o padrão SystemV. O Padrão SystemV é constituído em alguns "pedaços". Na verdade, estes pedaços são camadas que podemos utilizar para "dizer" quais serviços vão iniciar a partir do boot do sistema.

O processo de inicialização é feita da seguinte maneira, após o Kernel estar carregado na memória RAM, carregar os módulos, também os dispositivos que estão declarados dentro do arquivo "/etc/fstab", junto com seus respectivos dispositivos que estão declarados em /dev. Após estas etapas temos o inicio do carregamento dos serviços no sistema.

Então, na hora que o boot está sendo feito, antes dos serviços serem iniciados, temos que conhecer um diretório muito importante para o sistema:

```
#  cd  /etc/init.d

#  ls  -l
```

Dentro do diretório "/etc/init.d/" ficam todos os daemons. Um "daemon" é um script que é utilizado para dar "start" e "stop" em um serviço, outros aceitam o parâmetro "reload". Por exemplo:

```
#  /etc/init.d/exim4 stop
```

Nós estamos parando o serviço "Exim4", que é um serviço que trabalha na porta "25 SMTP", ele é um servidor de emails padrão da distribuição Debian.

Então podemos concluir que todos os scripts responsáveis por iniciarem ou pararem um determinado serviço ficam localizados dentro diretório "/etc/init.d/", ele é um repositório de daemons.

Quando o sistema é inicializado, o SystemV é ativado. Então o primeiro arquivo que será lido é o "/etc/inittab". Este arquivo armazena em qual "runlevel" o sistema irá inicializar:

```
#  head  /etc/inittab
```

Depois que ele lê qual é o nível de inicialização ou runlevel que o sistema irá inicializar na linha:

```
id:2:initdefault:
```

No padrão de inicialização SystemV, encontramos cinco níveis (runlevels):

- **S →** Carrega os serviços essenciais para o sistema;
- **0 →** Finaliza todos os serviços e desliga;
- **1 →** Carrega os serviços em modo mono-usuário;
- **2 – 5 →** Carrega os serviços em modo multi-usuário;
- **6 →** Finaliza todos os serviços e reinicia.

Na linha que estamos visualizando no arquivo "/etc/inittab", podemos dizer que nosso runlevel é o 2. Não só o runlevel do exemplo é 2, mas o nível de inicialização padrão do Debian é o 2.

Então depois de ler o arquivo "/etc/inittab", o sistema carrega os serviços que estão dentro do diretório "/etc/rcS.d/", este diretório sempre será carregado, pois ele

armazena quais são os serviços essenciais do sistema.

Por exemplo, nome da máquina, domínio, vamos entrar neste diretório e conhecer sua estrutura:

```
#  cd  /etc/rcS.d/
```

Quando listarmos seu conteúdo, vamos reparar muitos links apontando para o diretório “/etc/init.d”:

```
#  ls  -l
```

Ou seja, todos os scripts ficam dentro do diretório “/etc/init.d” como havíamos visto acima. Isso é muito bom, pois sabemos que o script sempre irá estar dentro deste diretório.

Após ele carregar os serviços essenciais, através do arquivo “/etc/inittab” nós podemos perceber que o nível de inicialização que estamos trabalhando é o 2, para podermos ver isso com agilidade, podemos executar o comando:

```
#  runlevel
```

Isso irá retornar o número 2, então após o carregamento dos scripts essenciais que ficam localizados no diretório “/etc/rcS.d/”, ele irá executar os scripts que estão dentro do diretório “/etc/rc2.d”, vamos visualizar:

```
#  cd  /etc/rc2/d

#  ls  -l
```

Podemos verificar que eles são links e também apontam para o diretório "/etc/init.d". Repare que temos scripts que começam a letra S ou K:

- **S** → Sinal de "start", o script será inicializado;
- **K** → Sinal de "stop", o script será finalizado.

Eles também recebem números logo em seguida, por exemplo no caso do exim4, o nome dele é declarado como S20exim4, então este serviço será o vigésimo a ser inicializado no boot.

Pois então, vamos fazer um teste. Agora que conhecemos como funciona o modo com que o sistema faz para inicializar os serviços, vamos observar o que acontece quando "brincamos" com os runlevels já em execução.

Verifique o runlevel que estamos utilizando:

```
#  runlevel
```

Vamos mudar para o nível 1, observe:

```
# init 1
```

O comando init é responsável por mudanças de runlevel, veja:

```
#  runlevel
```

Verifique que estavamos no 2 agora estamos no 1, podemos trocar novamente para 2:

```
#  init  2
```

Por exemplo, se quisermos desligar a máquina:

```
# init 0
```

Mas se quisermos reiniciar a máquina:

```
# init 6
```

Os diretórios deles, são respectivamente:

```
# cd /etc/rc0.d/ ; ls -l

# cd /etc/rc6.d/ ; ls -l
```

Para inicializar um serviço podemos passar o PATH completo, ou colocar:

```
# invoke.rc exim4 start
```

No caso do Red Hat, podemos utilizar:

```
# service postfix start
```

Quando iremos incluir um novo script de inicialização no sistema, ele precisa estar dentro do diretório "/etc/init.d", no Red Hat:

```
# chkconfig --add exim4
```

Capítulo 2

Desligando e reiniciando o sistema

2.1. Objetivos

- Desligar o sistema de modo seguro;
- Reiniciar o sistema de modo seguro;
- Utilizar os principais comandos do sistema.

2.2. Mãos a obra

Você alguma vez viu alguém desligando ou reiniciando a máquina puxando o cabo, com o dedo, muitas vezes presenciamos estas e outras cenas. Para que seu computador ou os servidores não tenham problemas, podemos fazer isto de modo seguro.

Então, para reiniciar a máquina de modo seguro, podemos utilizar o comando:

```
#  shutdown  -r  now
```

Essa opção “now” significa agora, então podemos reiniciar a máquina agora. O comando “shutdown” também aceita outros argumentos. Podemos além de dizer “now”, podemos dizer os minutos que podem levar para que isto aconteça. Por exemplo, para desligar a máquina daqui 10 minutos:

```
#  shutdown  -r  10
```

Além do comando “shutdown” para reinicializar a máquina, temos também o comando:

```
#  reboot
```

Este comando guarda alguns detalhes. Quando utilizamos o comando “reboot” o log do arquivo “/var/log/wtmp” armazena que a máquina foi reiniciada.

Verifique o tipo de arquivo que existe no sistema:

```
#  file  /var/log/wtmp
```

Este arquivo de log na verdade, é a saída do comando:

```
#  last
```

Este comando, como você pode reparar mostra os horarios que a máquina foi reiniciado, desligada e logins que foram feitos no ultimo mês.

```
root     pts/0        192.168.0.82     Tue May 25 11:43   still logged
root     tty1                          Tue May 25 11:42   still logged
root     tty1                          Tue May 25 11:42 - 11:42
reboot   system boot  2.6.26-2-686     Tue May 25 11:42 - 11:57
root     pts/0        192.168.0.82     Tue May 25 10:33 - 10:34
reboot   system boot  2.6.26-2-686     Fri May 21 10:47 - 16:19
root     pts/0        192.168.0.184    Fri May 21 10:45 - down
root     tty1                          Fri May 21 10:44 - down
root     tty1                          Fri May 21 10:44 - 10:44
root     pts/0        192.168.0.82     Tue May 18 14:39 - crash
```

Podemos reiniciar a máquina, sem que seja feito o log, assim:

```
#  reboot  -d
```

Com essa opção “-d” não será gravado nenhum registro no “wtmp”.

Mas ao listarmos o comando “reboot”, verificamos que ele é um link para o comando “halt”:

```
#  ls  -l  /sbin/reboot

lrwxrwxrwx 1 root root 4 Abr 15 16:08 /sbin/reboot -> halt
```

O comando “halt” é utilizado para desligar a máquina, e também trabalha com o “wtmp”, a opção “-d” do comando “halt” não envia o aviso para o “wtmp”.

Para desligar a máquina:

```
#  halt
```

Utilizando o comando shutdown, podemos trabalhar de forma similar. Para desligar a máquina agora:

```
#  shutdown  -h  now
```

Para desligar daqui 10 minutos:

```
#  shutdown  -h 10
```

Em ambas as opções podemos mandar uma mensagem dizendo algo para os usuários, esta mensagem é enviada via console texto:

```
#  shutdown  -h  10  Estamos Desligando o Servidor
```

Mas se por uma emergência, for necessário cancelar o comando shutdown, tanto com o "-r" do reboot, como no "-h" do halt, podemos enviar o sinal:

 # shutdown -c

Shutdown cancelled.

Podemos enviar uma mensagem também:

```
#  shutdown  -c  Agora não, estou fazendo backup
```

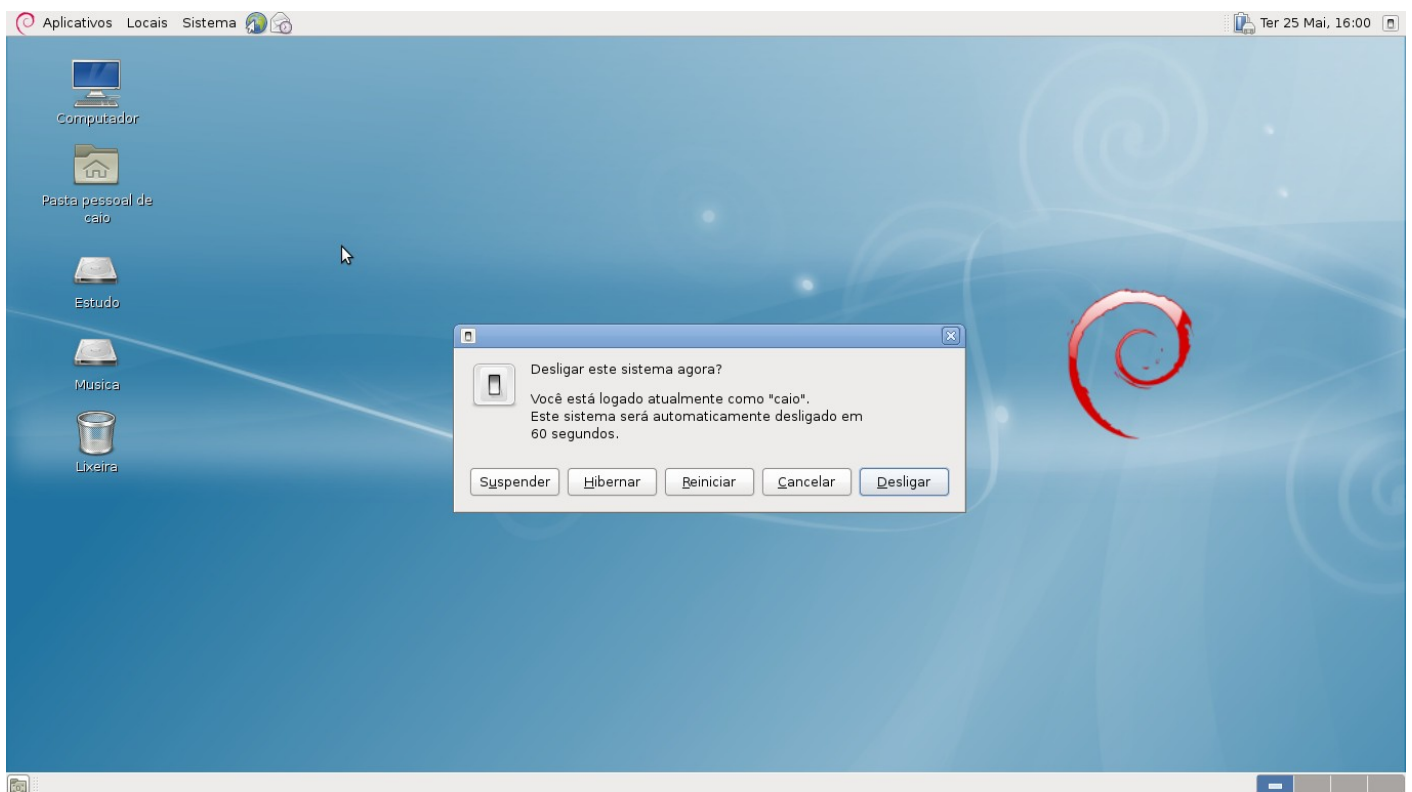
A saída do comando será mais ou menos assim:

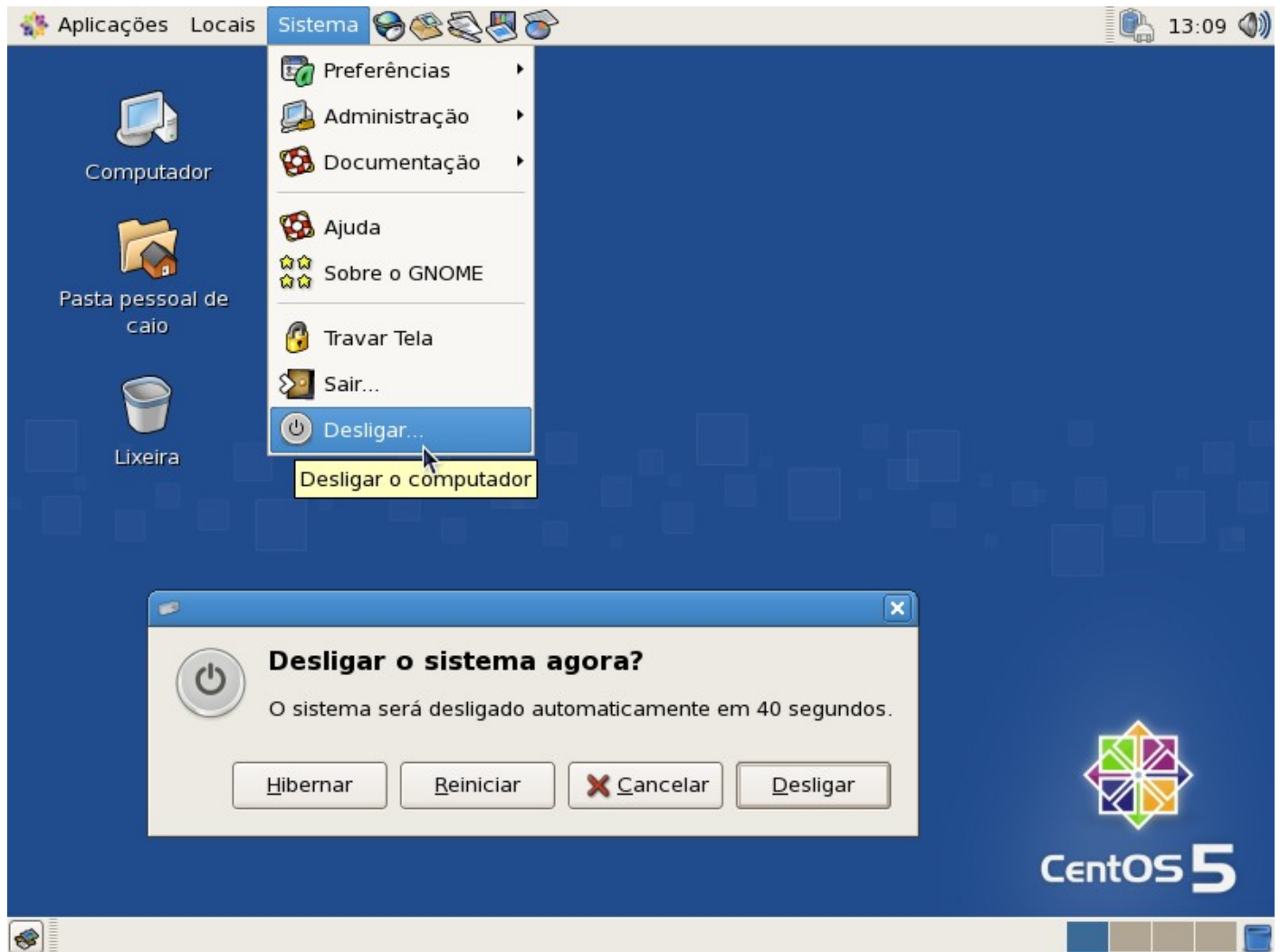
```
Shutdown cancelled.

Broadcast message from root@lenny (pts/0) (Tue May 25 12:43:44 2010):

Agora não, estou fazendo Backup
```

Poxa, depois de uma explicação como essa, nunca mais dê um "dedoviske" ou "dedada", tire o cabo do servidor. Utilize os comandos corretos para que nunca ocorra problemas de recuperação do sistema.

No modo gráfico não poderia ser diferente. No Debian, por exemplo:

Esta é a nossa "distro" oficial. Mas não podemos esquecer que as pessoas são livre para utilizar o que bem entenderem.

No CentOS:

Trabalhando como o modo gráfico, é possivél utilizar programas para gerenciar o boot do sistema:

Kshutdown (Para KDE)
Gshutdown (Para GNOME)

Capítulo 3

Gerenciando

3.1. Objetivos

- Inicializar e parar serviços com ferramentas;
- Deixar scripts incivilizáveis.

3.2. Troubleshooting

Imagine a seguinte situação: Seu chefe pede pra você colocar o script do firewall para inicializar a partir do boot. Como você faria isso?

Lembrando do que estudamos acima, vimos que o repositório de todos os “daemons” do sistema fica dentro:

```
# cd /etc/init.d
```

Dentro dele vamos criar um arquivo e atribuir permissão de execução ao arquivo:

```
# touch firewall

# chmod +x firewall
```

Estamos utilizando o “runlevel 2”, então podemos ver que a simples existência do arquivo “firewall” dentro do diretório “/etc/init.d/” não é suficiente para ele se tornar incivilizável.

```
# runlevel

# cd /etc/rc2.d

# ls -l
```

Podemos incluir ele no boot do sistema assim:

```
# update-rc.d firewall defaults
```

Repare a saída do comando:

```
Adding system startup for /etc/init.d/firewall ...
   /etc/rc0.d/K20firewall -> ../init.d/firewall
   /etc/rc1.d/K20firewall -> ../init.d/firewall
   /etc/rc6.d/K20firewall -> ../init.d/firewall
   /etc/rc2.d/S20firewall -> ../init.d/firewall
   /etc/rc3.d/S20firewall -> ../init.d/firewall
   /etc/rc4.d/S20firewall -> ../init.d/firewall
   /etc/rc5.d/S20firewall -> ../init.d/firewall
```

Por padrão, ele diz que nos runlevels onde encontramos serviços correntes, será o vigésimo script a ser “inicializado” quando a letra por “S”, ou “encerrado” quando a letra for “K”.

Podemos mudar isso, então poderíamos executar:

```
#  update-rc.d  firewall  defaults  99  30
```

Repare a saída do comando:

```
Adding system startup for /etc/init.d/firewall ...
   /etc/rc0.d/K30firewall -> ../init.d/firewall
   /etc/rc1.d/K30firewall -> ../init.d/firewall
   /etc/rc6.d/K30firewall -> ../init.d/firewall
   /etc/rc2.d/S99firewall -> ../init.d/firewall
   /etc/rc3.d/S99firewall -> ../init.d/firewall
   /etc/rc4.d/S99firewall -> ../init.d/firewall
   /etc/rc5.d/S99firewall -> ../init.d/firewall
```

Agora será o nonagésimo script a ser “inicializado” quando a letra por “S”, e quando for “encerrado” quando a letra for “K” será o trigésimo.

Mas vamos remover os links, e fazer de outra maneira:

```
# update-rc.d -f firewall remove
```

Lembre-se que só está funcionando porque o suposto "script firewall" se encontra dentro do diretório "/etc/init.d".

Vamos utilizar um utilitário no Debian que serve para trabalhar com os scripts. Para isso vamos utilizar o comando "aptitude":

```
# aptitude install rcconf
```

Antes de testarmos vamos remover o script do "CRON" dos runlevels, para nossos testes:

```
# update-rc.d -f cron remove
```

O cron é o serviço de agendador de tarefas do sistema. Não esqueçam que temos dois tipos, temos o agendamento por usuário e feito pelo sistema. Cuidado para não testar isso em um ambiente em que esteja rodando alguma coisa importante no exato momento em que retirar o daemon.

Verifique os trabalhos agendados por usuários e pelo sistema:

```
# crontab -l

# crontab -l usuario

# cat /etc/crontab
```

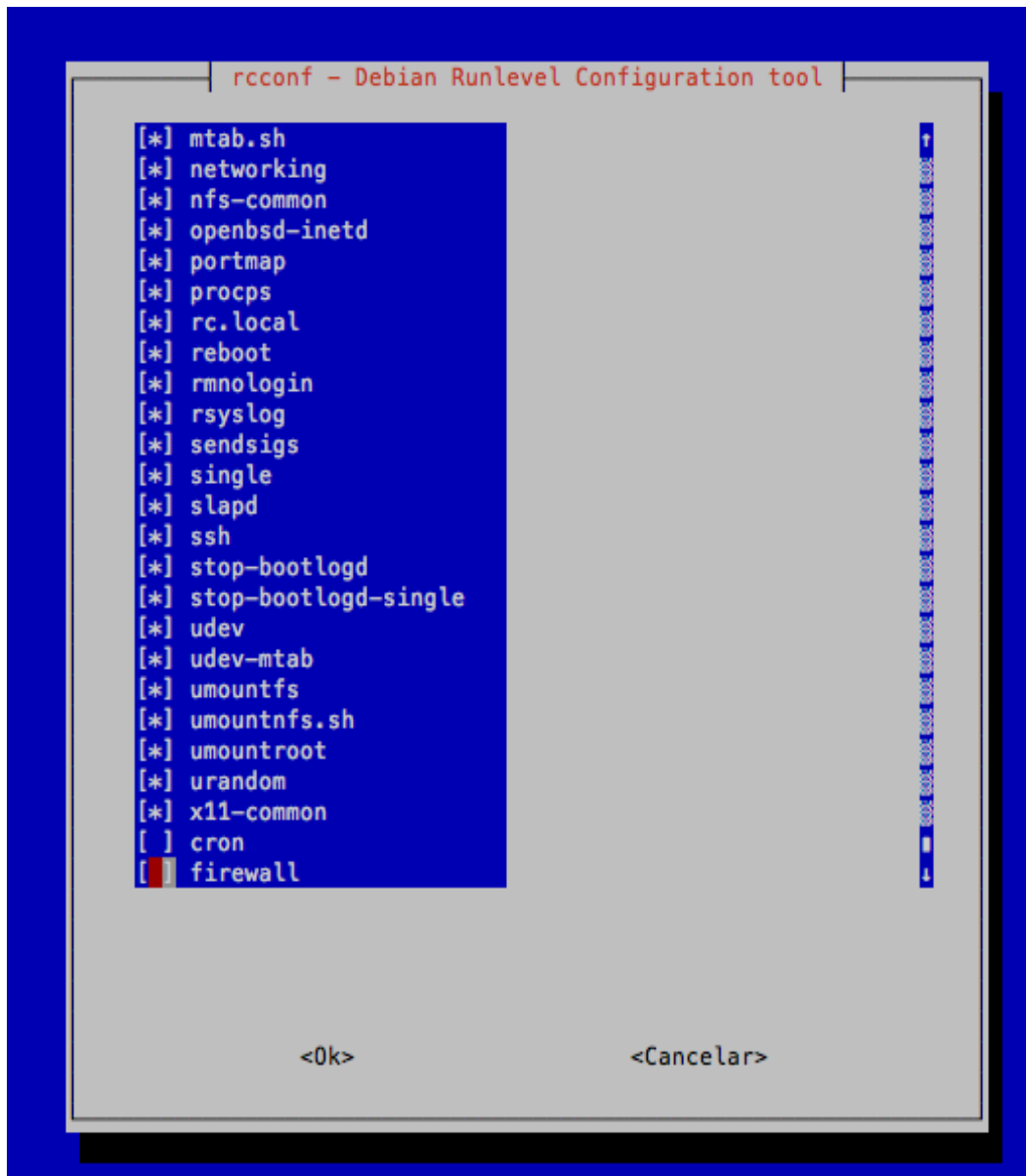
Agora iremos trabalhar com o "rcconf". Para executar:

 # rcconf

Ele irá abrir sua interface, que nada mais é do que todos os scripts que estão dentro do diretório "/etc/init.d":

```
rcconf - Debian Runlevel Configuration tool

[*] mtab.sh
[*] networking
[*] nfs-common
[*] openbsd-inetd
[*] portmap
[*] procps
[*] rc.local
[*] reboot
[*] rmnologin
[*] rsyslog
[*] sendsigs
[*] single
[*] slapd
[*] ssh
[*] stop-bootlogd
[*] stop-bootlogd-single
[*] udev
[*] udev-mtab
[*] umountfs
[*] umountnfs.sh
[*] umountroot
[*] urandom
[*] x11-common
[ ] cron
[ ] firewall

        <Ok>                <Cancelar>
```

*Repare que o cron e o firewall não estão marcados por um **
Isso significa que não estão vinculados a nenhum runlevel.

Vamos marcar o firewall com * para testar. Marque usando a tecla “espaço”, aperte o “tab” e clique em “OK”. Após isso, verifique onde ele está:

```
#  find  /etc  -name  S20firewall
```

Lembre-se que vinte é o número padrão atribuido. Então sua saída será:

```
/etc/rc4.d/S20firewall
/etc/rc3.d/S20firewall
/etc/rc5.d/S20firewall
/etc/rc2.d/S20firewall
```

Ou seja, qualquer um dos runlevels que iremos trabalhar, poderemos encontrar o sript do firewall.

Mas existe um comando que pode ser um pouco mais eficaz na hora de trabalhar com serviços em nosso dia-a-dia, vamos instalar:

```
# aptitude install sysv-rc-conf
```

Para executar este programa:

```
# sysv-rc-conf
```

Veja a descrição do pacote no site “sourceforge”:

http://sysv-rc-conf.sourceforge.net/

Irá aparecer:

```
 SysV Runlevel Config   -: stop service  =/+: start service  h: help  q: $

 service      1      2      3      4      5      0      6      S
 -----------------------------------------------------------------------$
 acpid       [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 atd         [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 bootlogd    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [X]
 cron        [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]
 exim4       [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 firewall    [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 halt        [ ]    [ ]    [ ]    [ ]    [ ]    [X]    [ ]    [ ]
 ifupdown    [ ]    [ ]    [ ]    [ ]    [ ]    [X]    [X]    [X]
 ifupdown-$  [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [X]
 killprocs   [X]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]
 module-in$  [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [X]
 mountover$  [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [X]
 networking  [ ]    [ ]    [ ]    [ ]    [ ]    [X]    [X]    [X]
 nfs-common  [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [X]
 openbsd-i$  [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 portmap     [ ]    [ ]    [ ]    [ ]    [ ]    [X]    [X]    [X]
 procps      [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [X]
 rc.local    [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 reboot      [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [X]    [ ]
 rmnologin   [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 rsyslog     [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 sendsigs    [ ]    [ ]    [ ]    [ ]    [ ]    [X]    [X]    [ ]
 single      [X]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]
 slapd       [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 ssh         [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 stop-boot$  [ ]    [X]    [X]    [X]    [X]    [ ]    [ ]    [ ]
 stop-boot$  [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [X]
 udev        [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [ ]    [X]
```

A grande vantagem é que podemos escolher em qual runlevel ou quais runlevels meu script irá executar. Em termos de segurança isso é muito bom, pois nas mudanças de runlevel, sabemos em quais irão funcionar.

Não esqueça que o runlevel padrão no Debian é o 2. E se encontra dentro do diretório "/etc/rc2.d/". Para alterar precisamos editar o arquivo "/etc/inittab" na linha "id:2:initdefault:".

Quando trabalhamos com “distros” baseadas em Red Hat, utilizamos:

 # ntsysv

No modo gráfico:

Aplicações Locais Sistema 10:39
Configuração dos Serviços
Arquivo Exibir Ações Editar Nível de Execução Ajuda
Salvar Reverter
Serviços em Segundo Plano | Serviços sob Demanda
Estes serviços são inicializados uma vez e rodam em segundo plano. Você pode especificar em quais níveis de execução eles serão inicializados.
Atualmente Executando no Nível de Execução: 5 Editando Nível de Execução: 5
Iniciar Parar Reiniciar
Descrição
This is a daemon for automatically switching network connections to the best available connection.
Status
NetworkManager está parado
NetworkManager
NetworkManagerDispatcher
acpid
anacron
apmd
atd
auditd
autofs
avahi-daemon
avahi-dnsconfd
bluetooth
Configuração dos Serviços